ELIOT: AN EMULATED IOT PLATFORM

Concept

Deploying a testbed of thousands of devices in a real environment just for prototyping purposes is time-consuming and expensive when compared to emulation.

ELIoT is an Emulated IoT Testbed that decouples software from hardware development.

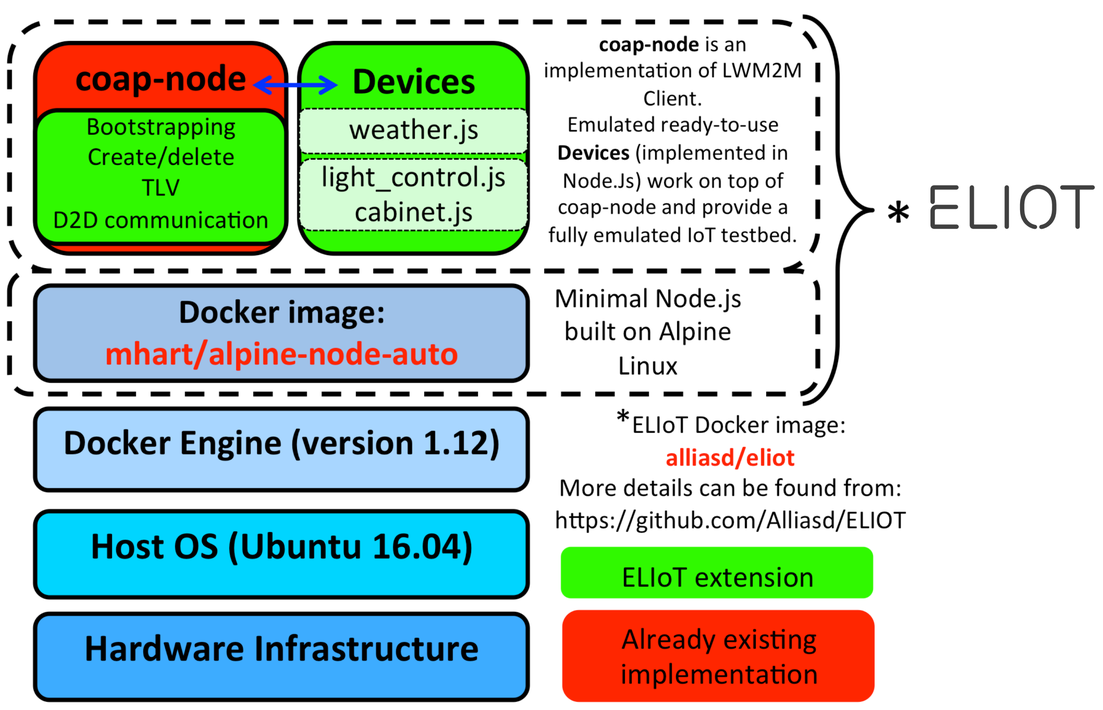
Background

ELIoT provides support for well-known IoT protocols:

The **Constrained Application Protocol (CoAP)** has been designed by the IETF as an universal transfer protocol with uniform identifiers for resources, methods for enabling resources interactions, resources discovery mechanisms, and security extensions.

On top of CoAP, the **Lightweight Machine-to-machine (LWM2M)** protocol has extended CoAP to a general purpose device management protocol.

LWM2M also defines a resource representation format. **LWM2M Objects** are used for device management operations on devices. **IPSO** further defines a set of general-purpose objects, which represent common type of sensors and actuators.

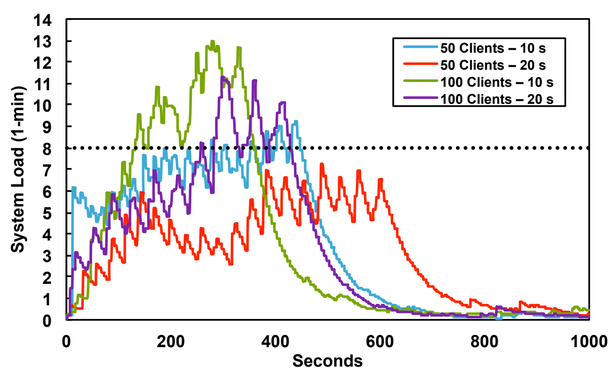


ELIoT has its own **docker** image - ***alliasd/eliot*4** - to store device applications, and configuration files needed to run the emulated devices. Each container represents an emulated device, that helps to characterize system resources and performance metrics of each emulated sensor.

ELIoT is built on top of a java-based LWM2M open source implementation called ***Leshan*** for the server side, and ***coap-node***, which is a nodejs implementation of LWM2M for the client side.

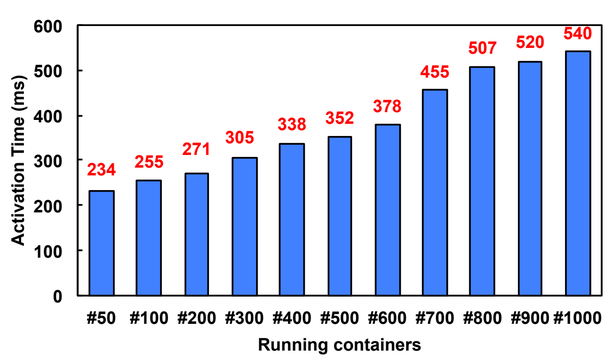
Evaluation

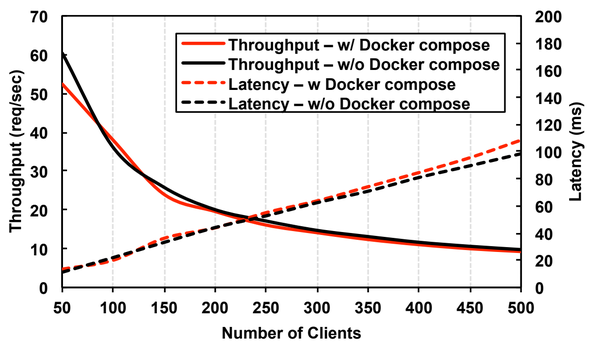
The European Telecommunications Standards Institute (ETSI) has defined a set of requirements for the public lighting scenario. Based on it ELIoT reproduces an IoT use case referred as *public lighting scenario*. The system emulates lighting points and a control cabinet. We also evaluate LWM2M's functionality in accordance to the COAP and LWM2M specifications.



1-minute load average of the host machine over 1000s. 1023 containers with various scheduling mechanisms.

Containers' activation time, increasing the device allocation by 100 until 1000 devices.





Latency increases linearly, while the throughput decreases fast from 50 to 150 clients and then slowly.

Completely stable performance requires around 14 MB of memory per container. To exceed the 1023 containers, we needed more memory and processing power. For that reason, we repeated the scalability test on an **Azure** cloud environment with 50 VMs and docker in host mode.

We used a docker bridge network with one dedicated VM for the leshan server and about 150 containers per VM.
In total we run **7123 devices** in the network, activation time was less than 2 minutes.

Conclusion

This work has already proven valuable to identify design issues with the LWM2M protocol itself, namely on the registration and discovery interfaces as well as the limitations of the object model it uses.

References

https://github.com/Alliasd/ELIoT

{ alli.a.makinen, jaime.jimenez, roberto.morabito } @ericsson.com